Voice Accent Transfer Using Recurrent Neural Networks on Spectrograms

Henry F. Wang
Department of Computer Science
Stanford University
henryfw@stanford.edu

Abstract

We investigate transferring one voice accent to another voice accent using a recurrent neutral network. We start by synthetically generating spectrograms of single words spoken in both American and British voices. The spectrogram image pairs are treated as time-series data and trained in a two-layer recurrent neural network. We present the successful results of our research.

1 Introduction

The goal of this project is to be able to transfer speech from the American accent to the British accent by using a RNN model. Input data are spectrogram images of computer synthesized words in American accent. The training labels are corresponding spectrogram images of computer synthesized words in British accent. We use a two-layer recurrent neural network with two fully connected layers at each time step to generate British accented spectrogram images from American accented spectrogram images.

A spectrogram is a visual representation of speech. The x-axis represents time, the y-axis represents pitch, and the value at the x-y coordinate represents the amplitude. The Griffin-Lim algorithm is used to convert the spectrogram back to audio waveform.

2 Related Work

Speech generation using custom voices has been explored in WaveNet [1] by using stacked dilated causal convolutions at the waveform level. The WaveNet model is able to capture the characteristics of over a hundred speaks in one model. One drawback of WaveNet is that it requires pre-processed linguistic features to generate speech.

The state-of-the-art Deep Voice [2] is a standalone system for text-to-speech, consisting of 5 models: a segmentation model for locating phoneme boundaries, a grapheme-to-phoneme conversion model, a phoneme duration prediction model, a fundamental frequency prediction model, and an audio synthesis model that takes WaveNet even further by requiring fewer parameters and offering faster training.

Tacotron [2] is another standalone system for speech generation. It does not require annotated phonemes for training. Tacotron uses RNN based encoder and decoder to generate spectrograms.

Amy Bearman, et al. explored accent conversion using neural networks [4]. Mel-frequency cepstral coefficients (MFCC) are extracted from parallel utterances from American, Indian, and Scottish English and used in a two-layer dense neural network. The paper explored using RNN models, but did not find promising results.

The style transfer model is used by Anthony Perez, et al. for speech [5]. The paper used spectrograms and reported partial success.

In most of the papers above, the individual characteristic of the input voice is lost, because the intermediate MFCC or phoneme representations are sound independent. We hope that by skipping this intermediate representation, we can maintain some of the input voice's individual characteristic.

3 Dataset and Features

The dataset is made of pairs of single word spectrograms that are made from computer generated waveform audio files using voices in American accent and British accent. The American accent version is the input, and the British accent version is the label. Due to computational constraints, the full spectrograms of 300x257 are cropped to 50x50 for initial experimentations and 150x50 in the final model results.

The values in the spectrograms are normalized to 0 and 1.0 and used as features in the time-series dataset for training seq-to-seq RNN models.

Words are mined from over 90,000 IMDB movies reviews [6] with more frequent words repeated up to 5 times. The total dataset generated has 350,000 pairs of spectrogram images in randomized order. For initial experimentations, 5000 pairs are used for training and validation. For the final model, 40,000 pairs are used for training and 5000 pairs are used for validation.

4 Methods

The two-layer recurrent neural network setup is usually used as the encoder and decoder layers in language translation. We use it here, but altered with many skip connections at different layers. We tested both GRU and LSTM models in similar configurations. The two-layer LSTM model with two fully connected layers is described in Figure 1. One time step is first passed to a LSTM layer. The state of the first LSTM is concatenated with the input and then passed to the second LSTM layer. Then, both of the LSTM states and the input are concatenated and passed to a fully connected layer. Finally the prediction is generated by another fully connected layer with the size of the output image height.

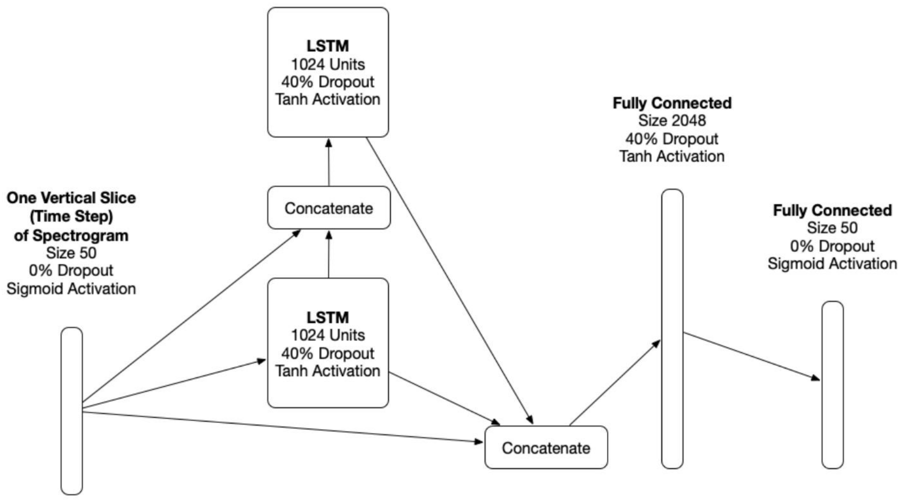

Figure 1: Network architecture of RNN model.

The loss function is the mean absolute error or L1 error, which is calculated by the sum of the absolute differences at each image pixel. The loss function is given below. We did not use L2 error because of underflow problems from using the 16-bit float data type in the dataset.

$$\sum_{i=0}^{x} \sum_{j=0}^{y} abs\left(y_{ij} - \overline{y_{ij}}\right)$$

We use the Keras framework. We choose the training optimizer to be RMSprop based on the Keras documentation suggesting RMSprop for RNN [8]. Dropout of 40% is used on all layers except the final output to prevent overfitting. All layers use tanh activation except the last layer, which use the sigmoid activation since the image values are normalized to 0 and 1.0.

5 Experiments/Results/Discussion

We used 0.001 as the learning rate, 1024 as the number of hidden units, and tanh activation functions because they proved to work well. The mini-batch size of 128 is chosen to maximize the usage of GPU memory. The primary metric is accuracy.

Different variations of the model architecture were tried in the initial experimental phase with 5000 samples. The result of the L1 errors after 100 epochs are shown in Table 1.

LSTM with All Skip Connections (Figure 1)	0.0304
GRU with All Skip Connections	0.0323
LSTM with No Skip Connection	0.0622
LSTM with Only Skip Connection to 1st Fully Connected Layer	0.0378
LSTM with Only Skip Connection to 2nd LSTM	0.0347
LSTM with Only 1 LSTM Layer and All Skip Connections	0.0622
LSTM with Only 1 Fully Connected Layer and All Skip Connections	0.0551

Table 1: L1 Errors from various modifications to Figure 1 architecture.

From Table 1, we tried to simplify the model by removing the skip connections at various layers and trying GRU models. None performed as well as the model from Figure 1. We trained the final model architecture from Figure 1 on 40,000 samples for 400 epochs for over 40 hours on a NVIDIA 1070 GTX. We stopped training when the validation loss flattened to prevent overfitting.

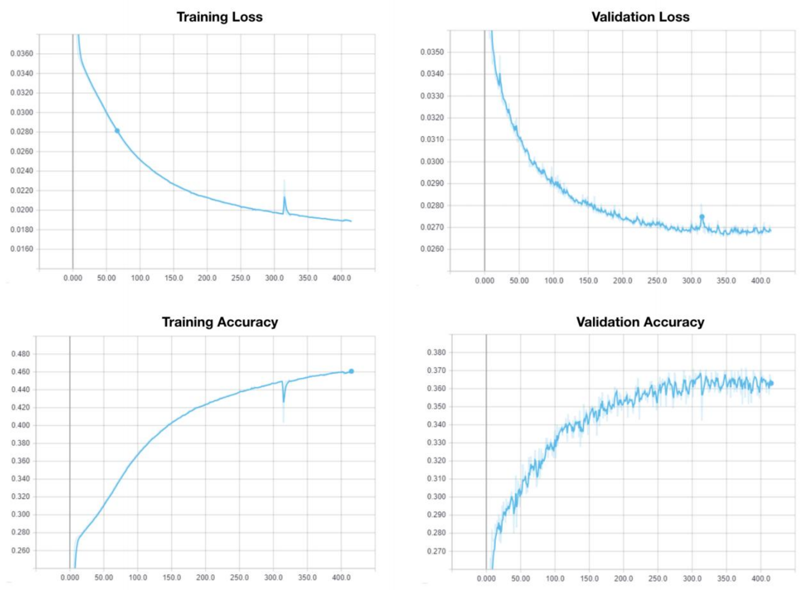

Figure 2: Final model training and validation metrics.

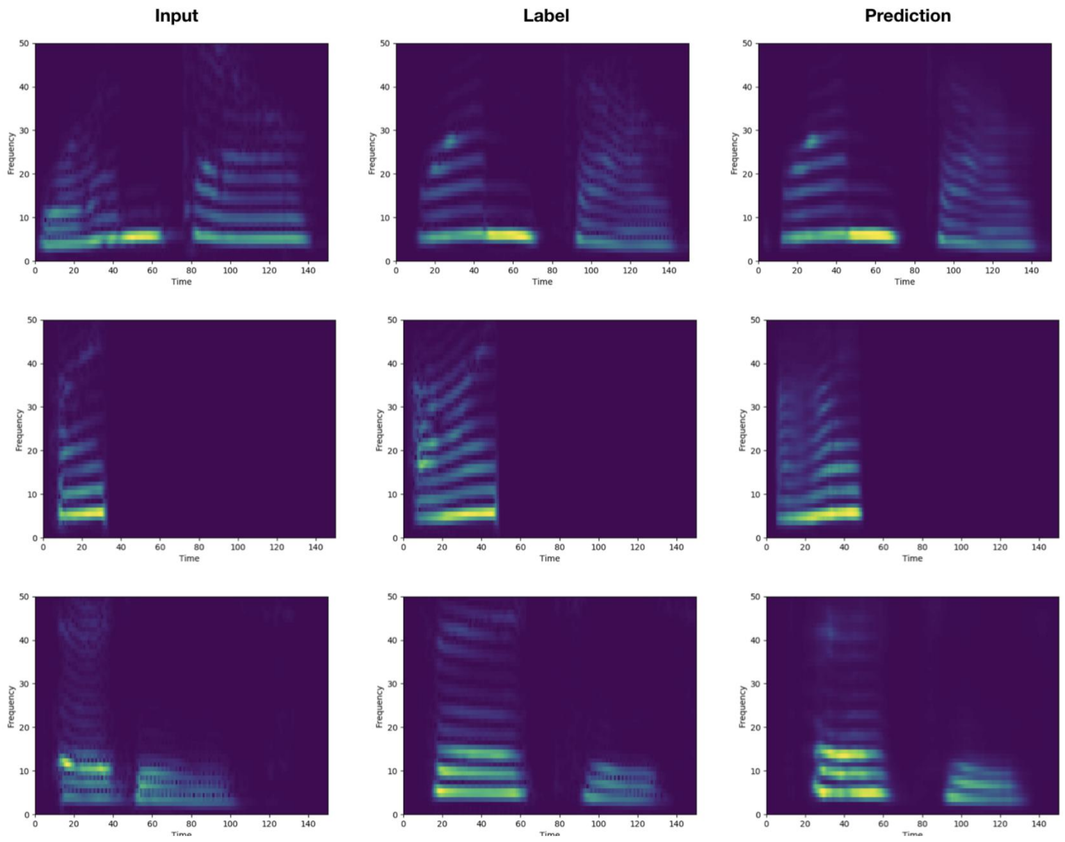

Figure 3: Sample test set results from final model.

Figure 3 shows three example of the final model from using a test set. The first column is the American accent spectrogram, the second column is the British accent spectrogram, and the third column is the prediction from our model. The model did moderately well for learning the difference between the spectrograms in two accents.

Almost all the layers received the input in addition to the immediate previous layer. These skip connections help the back-propagation steps during training, especially when using tanh activations. From Table 1, we can see that without any skip connections, the loss becomes twice large. Also, from Table 1, we can see that having two LSTM layers and two fully connected layers are critical.

From experimentations, we find that dropouts are very helpful in preventing overfitting with respect to the validation loss.

6 Conclusion/Future Work

Recurrent neural networks are able to translate spectrograms between two synthesized accents. The highest performing architecture from our project requires two LSTM layers, two fully connected layers, and skip connections. We hypothesize that the layers are needed to learn the shifting of pixels through time, and that the skip connections helped in speeding up back propagation.

This project only trains one single word at a time within a fixed temporal window. This means pre-processing and post-processing will be required to chop sentences into words, which can be future work.

Due to time and resource constraints, the full spectrograms of 300x257 are cropped. Future work could be done on the entire image. While this project only used 40,000 samples to train, future work can use all of the 350,000 generated samples. Data augmentation can also be performed by speaking each word at different speeds. This project generated speech at 175 words per minute.

In the future, greater vocabulary and additional synthetically generated voice pairs can be used. This would enable the model to be more life like. With a more life like model, subjective tests using real American accented human speakers can be used as the input.

7 Contributions

Henry Wang completed this project as an individual. Source code is located at https://github.com/henryfw/cs-230.

References

1) Oord, A., et al. (2016) WaveNet: A Generative Model for Raw Audio. https://arxiv.org/abs/1609.03499

2) Arik, S. O., et al. (2017) Deep Voice: Real-time Neural Text-to-Speech. https://arxiv.org/abs/1702.07825

3) Wang, Y., et al. (2017) Tacotron: Towards End-to-End Speech Synthesis. https://arxiv.org/abs/1703.10135

4) Bearman, A., et al. Accent Conversion Using Artificial Neural Networks. http://web.stanford.edu/class/cs224s/reports/Amy_Bearman.pdf

5) Perez, A., et al. Style Transfer for Prosodic Speech. http://web.stanford.edu/class/cs224s/reports/Anthony_Perez.pdf

6) Large Movie Review Dataset. http://ai.stanford.edu/~amaas/data/sentiment/

7) Keras Optimizers. https://keras.io/optimizers/